

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 06-266554

(43)Date of publication of application : 22.09.1994

(51)Int.Cl.

G06F 9/30

G06F 7/00

G06F 15/72

(21)Application number : 05-056792

(71)Applicant : HITACHI LTD

(22)Date of filing : 17.03.1993

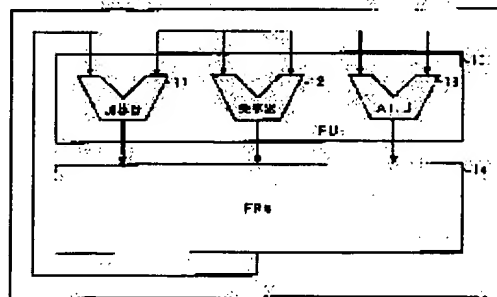
(72)Inventor : SUZUKI KATSUNORI
FUJITA MAKOTO
KOGA KAZUYOSHI
FUJII HIDEKI

(54) MICROPROCESSOR

(57)Abstract:

PURPOSE: To provide a microprocessor suitable for a graphic system capable of remarkably accelerating a packing processing and an unpacking processing and to provide the graphic system to which the microprocessor is applied.

CONSTITUTION: This microprocessor for performing the geometrical processing of the graphic system is provided with a floating register, an adder 11, a multiplier 12 and an ALU 13 for performing a bit arithmetic processing inside a floating (floating point real number) arithmetic unit. Thus, since the microprocessor can perform the bit arithmetic processing with the floating register, the packing processing and the unpacking processing can be performed with the floating register at a high speed.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of

BEST AVAILABLE COPY

rejection]

[Kind of final disposal of application other than
the examiner's decision of rejection or
application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's
decision of rejection]

[Date of requesting appeal against examiner's
decision of rejection]

[Date of extinction of right]

Copyright (C); 1998,2003 Japan Patent Office

(19)日本国特許庁(JP)

(12) 公開特許公報(A)

(11)特許出願公開番号

特開平6-266554

(43)公開日 平成6年(1994)9月22日

(51)Int.Cl.⁵

G 0 6 F 9/30
7/00
15/72

識別記号 庁内整理番号
3 5 0 E 9189-5B

F I

技術表示箇所

9188-5B

G 0 6 F 7/ 00

1 0 1 W

審査請求 未請求 請求項の数4 O L (全 13 頁)

(21)出願番号 特願平5-56792

(22)出願日 平成5年(1993)3月17日

(71)出願人 000005108

株式会社日立製作所

東京都千代田区神田駿河台四丁目6番地

(72)発明者 鈴木 克徳

茨城県日立市大みか町七丁目1番1号 株
式会社日立製作所日立研究所内

(72)発明者 藤田 良

茨城県日立市大みか町七丁目1番1号 株
式会社日立製作所日立研究所内

(72)発明者 古賀 和義

茨城県日立市大みか町七丁目1番1号 株
式会社日立製作所日立研究所内

(74)代理人 弁理士 小川 勝男

最終頁に続く

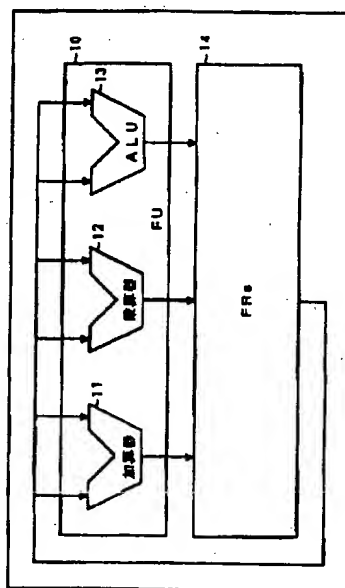
(54)【発明の名称】 マイクロプロセッサ

(57)【要約】

【目的】本発明は、pack処理、pnpack処理を著しく高速化することができるグラフィックス・システムに適したマイクロプロセッサ、及びそれを適用したグラフィックス・システムを提供することを目的としている。

【構成】前記目的を達成するために、本発明のグラフィックス・システムの幾何処理を行うマイクロプロセッサは、floating(浮動小数点実数)演算ユニット内に、floatingレジスタと、加算器と、乗算器と、bit演算処理を行うALUを有する。これにより、本発明のマイクロプロセッサは、floatingレジスタ間でbit演算処理を行うことができるので、pack処理、unpack処理をfloatingレジスタ間で高速に行うことが可能である。

図 1



【特許請求の範囲】

【請求項1】グラフィックス・システムの幾何処理を行うマイクロプロセッサにおいて、

floating (浮動小数点実数) 演算ユニット内に、floatingレジスタと、加算器と、乗算器と、bit演算処理を行うALUを有し、(1) floatingレジスタにある1画素のデータを構成する各成分を示す複数のfloating型のデータを加算器によりfloating型からinteger(整数)型へ変換し、(2) 前記変換結果である各成分を示す複数のinteger型のデータの有効なbit部分をALUにより1つの画素データに変換するpack処理をfloatingレジスタ間で行うことを特徴とするマイクロプロセッサ。

【請求項2】グラフィックス・システムの幾何処理を行うマイクロプロセッサにおいて、

floating (浮動小数点実数) 演算ユニット内に、floatingレジスタと、加算器と、乗算器と、bit演算処理を行うALUを有し、(1) floatingレジスタにある複数の成分から構成される1つの画素データをALUにより各成分を示す複数のinteger型のデータに変換し、(2) 前記変換結果である各成分を示す複数のinteger型のデータを加算器によりinteger型からfloating型へ変換するunpack処理をfloatingレジスタ間で行うことを特徴とするマイクロプロセッサ。

【請求項3】グラフィックス・システムの幾何処理を行うマイクロプロセッサにおいて、

floating (浮動小数点実数) 演算ユニット内に、floatingレジスタと、加算器と、乗算器と、bit演算処理を行うALUを有し、(1) floatingレジスタにある1画素のデータを構成する各成分を示す複数のfloating型のデータを加算器によりfloating型からinteger(整数)型へ変換し、(2) 前記変換結果である各成分を示す複数のinteger型のデータの有効なbit部分をALUにより1つの画素データに変換するpack処理をfloatingレジスタ間で行う命令を有することを特徴とするマイクロプロセッサ。

【請求項4】グラフィックス・システムの幾何処理を行うマイクロプロセッサにおいて、

floating (浮動小数点実数) 演算ユニット内に、floatingレジスタと、加算器と、乗算器と、bit演算処理を行うALUを有し、(1) floatingレジスタにある複数の成分から構成される1つの画素データをALUにより各成分を示す複数のinteger型のデータに変換し、(2) 前記変換結果である各成分を示す複数のinteger型のデータを加算器によりinteger型からfloating型へ変換するunpack処理をfloatingレジスタ間で行う命令を有することを特徴とするマイクロプロセッサ。

【発明の詳細な説明】

【0001】

【産業上の利用分野】本発明は、グラフィックス・システムに適したマイクロプロセッサ、及びそれを適用した

グラフィックス・システムに関する。更に詳細に言えば、グラフィックス・システムにおける画素データのpack処理とunpack処理、即ち、(1) 1画素のデータを構成する各成分を示す複数のfloating型のデータをinteger型へ変換し、前記変換結果である各成分を示す複数のデータの有効なbit部分を1つの画素データに変換する処理と、(2) 複数の成分から構成される1つの画素データを各成分を示す複数のデータに変換するbit演算処理と、前記変換結果である各成分を示す複数のinteger型のデータをfloating型へ変換する処理の高速化に関する。

【0002】

【従来の技術】コンピュータ・グラフィックス・システムは、コンピュータの出力を図形として表示するものである。従来のグラフィックス・システムの図形表示方法は、「PEX Introduction and Overview (M.I.T., 1988, pp51-72)」に詳細が記載されている。

【0003】グラフィックス・システムは、図形の基準点の座標変換と、図形、光源、視点の位置、色などの情報より図形がどの様に見えるか光源計算を行い図形の基準点の色を算出する幾何処理と、図形の基準点の情報からその図形の内部の画素を内挿補間により1画素ずつ展開して描画するレンダリング処理を行い、ディスプレイに表示する内容をビットマップ形式で保持する画像メモリに書き込む。グラフィックス・システムは、一般に、画素データを図形の座標データXYZと色データRGB (Red, Green, Blue)で表現し、前記幾何処理ではfloating (浮動小数点実数) 型で計算し、前記レンダリング処理ではinteger (整数) 型で計算する。ここでは、説明を簡単にするために、画像メモリの画素データの色データRGBの各成分が各8bit、合計24bitであるグラフィックス・システムについて説明する。

【0004】前記の様に、floating型のr, g, bをinteger型のI r, I g, I bに変換し、各成分別々I r, I g, I bの有効な各8bitを1つの24bitのデータに変換して画像メモリの画素データを表現するため、グラフィックス・システムでは、RGB各成分を示す複数のfloating型のデータr, g, bを、RGB各成分を示す複数のinteger型のデータI r, I g, I bに変換するデータ型の変換処理と、前記変換結果であるI r, I g, I bの有効な各8bit部分を1つの画素データに変換するbit演算処理が画像メモリ書き込み時に必ず発生する。ここでは、この一連の処理をpack処理と呼ぶ。又、その逆変換処理、即ち、1つの画素データをRGB各成分を示す複数のinteger型のデータI r, I g, I bに変換するbit演算処理を行い、前記変換結果であるI r, I g, I bをRGB各成分を示す複数のfloating型のデータr, g, bに変換するデータ型の変換処理も発生する。ここでは、この一連の処理をunpack処理と呼ぶ。

【0005】次に、従来のpack処理とunpack処理について説明する。

【0006】まず、従来のpack処理について説明する。

【0007】図4に、従来のpack処理のアセンブリ言語によるプログラムとレジスタの状態の例を示す。

【0008】但し、%grfはinteger演算を行うためのgeneral (汎用) レジスタ、%frfはfloating演算を行うためのfloatingレジスタ、/*...*/はコメント (注釈) である。

【0009】(1) 初めに、L402, 403, 404 10
では、初期状態として、RGBの各成分を示す複数のfloating型のデータr, g, bがfloatingレジスタ16, 17, 18番にある(402, 403, 404)。

【0010】(r, g, bはfloating型, $0.0 \leq r, g, b \leq 255.0$)

(2) そして、L406, 407, 408では、floating型のデータをinteger型のデータに変換する命令fcvnxにより、前記データr, g, bをinteger型のデータ1r, 1g, 1bへ変換するデータ型の変換処理を順次3回行う(406, 407, 408)。

【0011】(1r, 1g, 1bはinteger型, $0 \leq 1r, 1g, 1b \leq 255$)

(3) 次に、bit演算処理を行うわけだが、一般にfloatingレジスタではbit演算処理を行うことができない。そこで、bit演算処理を行うために、L410, 411, 412では、floatingレジスタからgeneralレジスタへデータを転送する命令frtogrにより、前記データ1r, 1g, 1bをfloatingレジスタ16, 17, 18番からbit演算処理が可能なgeneralレジスタ16, 17, 18番へ転送するデータ転送処理を順次3回行う(410, 411, 412)。

(4) 最後に、前記データ1r, 1g, 1bの有効な各8bitを1つの画素データに変換するために、L414, 415, 416ではbit演算処理を行う命令depにより、generalレジスタ16, 17, 18番にある前記1r, 1g, 1bの有効な各8bitをgeneralレジスタ19番の各8bitの部分に設定するbit演算処理を順次3回行う(414, 415, 416)。

【0012】前記の様に従来のpack処理では、floatingレジスタでfloating型のデータr, g, bをinteger型のデータ1r, 1g, 1bへ変換した後、floatingレジスタでは一般にbit演算処理を行えないので、floatingレジスタからbit演算処理が可能なgeneralレジスタへ転送し、generalレジスタでbit演算処理を行い1つの画素データに変換する。このため、floatingレジスタからgeneralレジスタへのデータ転送処理と、更に、順次3回のbit演算処理が必要となり、pack処理に多大な時間を要するという問題を有する。

【0013】次に、従来のunpack処理について説明する。

【0014】図5に、従来のunpack処理のアセンブリ言語によるプログラムとレジスタの状態の例を示す。

【0015】(1) 初めに、L502では、初期状態として、RGBの各成分1r, 1g, 1bの各8bitを既にpack処理した画素データがgeneralレジスタ19番にある(502)。

【0016】(1r, 1g, 1bはinteger型, $0 \leq 1r, 1g, 1b \leq 255$)

(2) そして、L504, 505, 506ではbit演算処理を行う命令extruにより、前記1r, 1g, 1b各8bitをそれぞれgeneralレジスタ16, 17, 18番に設定するbit演算処理を順次3回行う(504, 505, 506)。

(3) 次に、データ型の変換処理を行うわけだが、一般にgeneralレジスタではデータ型の変換処理を行うことができない。そこで、データ型の変換処理を行うために、L508, 509, 510ではgeneralレジスタからfloatingレジスタへデータを転送する命令grtofrにより、前記データ1r, 1g, 1bをgeneralレジスタ16, 17, 18番からデータ型の変換処理が可能なfloatingレジスタ16, 17, 18番へ転送するデータ転送処理を順次3回行う(508, 509, 510)。

【0017】(4) 最後に、L512, 513, 514ではinteger型のデータをfloating型のデータに変換する命令fcvxfにより、floatingレジスタ16, 17, 18番にある前記データ1r, 1g, 1bをfloating型のデータr, g, bへ変換するデータ型の変換処理を順次3回行う(512, 513, 514)。

【0018】(r, g, bはfloating型, $0.0 \leq r, g, b \leq 255.0$)

前記の様に従来のunpack処理では、1つの画素データをgeneralレジスタでbit演算処理した後、generalレジスタからデータ型の変換処理が可能なfloatingレジスタへ転送し、floatingレジスタでinteger型のデータをfloating型のデータへ変換する。このため、generalレジスタからfloatingレジスタへのデータ転送処理と、更に、順次3回のbit演算処理を必要となり、unpack処理に多大な時間を要するという問題を有する。

【0019】

【発明が解決しようとする課題】前記の様に従来のpack処理では、floatingレジスタからgeneralレジスタへのデータ転送処理と、更に、順次3回のbit演算処理を必要とし、pack処理に多大な時間を要するという問題を有する。

【0020】又、従来のunpack処理では、generalレジスタからfloatingレジスタへのデータ転送処理と、更に、順次3回のbit演算処理を必要とし、unpack処理に多大な時間を要するという問題を有する。

【0021】従って、本発明は、前記問題を解決して、pack処理、unpack処理を著しく高速化することがで

きるグラフィックス・システムに適したマイクロプロセッサ、及びそれを適用したグラフィックス・システムを提供することを目的としている。

【0022】

【課題を解決するための手段】前記目的を達成するために、本発明のグラフィックス・システムの幾何処理を行うマイクロプロセッサは、floating（浮動小数点実数）演算ユニット内に、floatingレジスタと、加算器と、乗算器と、bit 演算処理を行うALUを有する。これにより、本発明のマイクロプロセッサは、floatingレジスタ間でbit 演算処理を行うことができるので、floatingレジスタ間でpack処理、unpack処理を高速に行うことが可能となる。

【0023】以上の様に構成すれば、グラフィックス・システムにおける画素データのpack処理、unpack処理を高速に行うことができるグラフィックス・システムに適したマイクロプロセッサを構成できる。又、それをグラフィックス・システムに適用することにより、画素データのpack処理、unpack処理を著しく高速に行うことができるグラフィックス・システムを構成できる。

【0024】

【作用】本発明のグラフィックス・システムの幾何処理を行うマイクロプロセッサは、pack処理において、floatingレジスタにあるfloating型のデータを加算器によりinteger 型のデータへ変換し、前記変換結果であるinteger 型のデータの有効なbit 部分をALUによりそのままfloatingレジスタでbit 演算処理を行い1つの画素データに変換する。この様に、本発明のマイクロプロセッサは、floatingレジスタからgeneral レジスタへのデータ転送処理を行わないので、pack処理を著しく高速化することが可能となる。

【0025】又、本発明のグラフィックス・システムの幾何処理を行うマイクロプロセッサは、unpack処理において、floatingレジスタにある複数の成分から構成される1つの画素データをALUによりそのままfloatingレジスタで各成分を示す複数のinteger 型のデータに変換するbit演算処理を行い、前記変換結果であるinteger型のデータを加算器によりfloating型のデータへ変換する。この様に、本発明のマイクロプロセッサは、general レジスタからfloatingレジスタへのデータ転送処理を行わないので、unpack処理を著しく高速化することが可能となる。

【0026】以上の様に、本発明のマイクロプロセッサは、グラフィックス・システムにおける画素データのpack処理、unpack処理を著しく高速化することができる。又、それを適用したグラフィックス・システムは、グラフィックス・システムにおける画素データのpack処理、unpack処理を著しく高速化することが可能となる。

【0027】

【実施例】以下、実施例を図面によって詳細に説明す

る。

【0028】図10は、本発明のグラフィックス・システムの一実施例を示す構成図である。

【0029】本発明のグラフィックス・システムは、幾何処理を行うマイクロプロセッサCPU（101）、メモリ（103）等を制御するコントローラ（102）、レンダリング処理を行う描画プロセッサ（104）、描画した画像を保持する画像メモリ（105）、画像を表示するディスプレイ（106）から構成される。

【0030】CPU（101）は、アプリケーションを実行し幾何処理を行い図形の基準点の画素データとグラフィックス・コマンド（描画コマンド）を発行し、コントローラ（102）を通して描画プロセッサ（104）に送出する。描画プロセッサ（104）は、図形の基準点の画素データとグラフィックス・コマンドから図形の内部の画素を内挿補間により1画素ずつ展開して描画するレンダリング処理を行い、ディスプレイに表示する内容をビットマップ形式で保持する画像メモリ（105）に書き込み、画像をディスプレイ（106）に表示する。

【0031】グラフィックス・システムは、画素データを、幾何処理ではfloating（浮動小数点実数）型で計算し、レンダリング処理ではinteger（整数）型で計算する。ここでは、説明を簡単にするために、画像メモリの画素データの色データRGBの各成分が各8bit、合計24bitであるグラフィックス・システムについて説明する。

【0032】まず、幾何処理を行うマイクロプロセッサCPU（101）について説明する。

【0033】図2は、本発明のマイクロプロセッサの一実施例を示す構成図である。

【0034】本発明のマイクロプロセッサは、マイクロプロセッサと外部を接続するSystem Interface（21）、命令キャッシュIC（22）、データキャッシュDC（23）、マイクロプロセッサの各ユニットを制御するControler（24）、integer 演算を行うためのgeneral レジスタGRs（25）、integer 演算を実際に行うinteger 演算ユニットIU（26）、floating演算を行うためのfloatingレジスタFRs（27）、floating演算を実際に行うfloating演算ユニットFU（28）から構成される。

【0035】本発明のマイクロプロセッサは、System Interface（21）を介して、命令とデータを読み込み、命令キャッシュIC（22）、データキャッシュDC（23）に格納し、命令に従ってControler がマイクロプロセッサの各ユニットを制御し、必要なデータがGRs（25）、FRs（27）に格納され、IU（26）、FU（28）が命令の処理を実行する。

【0036】次に、FU（28）の構成について詳細に説明する。

【0037】図1は、本発明のマイクロプロセッサのFUとFRsの一実施例を示す構成図である。

【0038】FU(10)は、加減算やデータ型の変換などの演算を行う加算器(11)、乗除算や平方根などの演算を行う乗算器(12)、pack処理、unpack処理に必要なbit演算処理を行うALU(13)から構成される。FU(10)は、命令に従って、FRs(14)のデータを読み込み、加算器(11)、乗算器(12)、ALU(13)で演算を行い、その結果のデータをFRs(14)に書き込む。次に、本発明のマイクロプロセッサにおける高速なpack処理とunpack処理について詳細に説明する。

【0039】まず、pack処理について説明する。

【0040】図6に、本発明のマイクロプロセッサにおける高速なpack処理のアセンブリ言語によるプログラムとレジスタの状態の例を示す。

【0041】(1)初めに、L602、603、604では、初期状態として、RGBの各成分を示す複数のfloating型のデータr、g、bがfloatingレジスタ16、17、18番にある(602、603、604)。

【0042】(r、g、bはfloating型、 $0.0 \leq r, g, b \leq 255.0$)

(2)次に、L607では、floating型のデータをinteger型のデータへ変換してbit演算処理を行う命令fcvfxdepにより、前記floatingレジスタ16番にあるデータrをinteger型のデータ1rへ変換するデータ型の変換処理を前記加算器(11)で行い、更に、前記データ1rの有効な8bitをfloatingレジスタ19番の所定の8bitの部分に設定するbit演算処理を前記ALU(13)で行う(607)。同様に、L608、609では、前記データg、bについて同様な処理を行う(608、609)。この様にして、floating型のデータr、g、bをinteger型のデータ1r、1g、1bに変換し、integer型のデータ1r、1g、1bの有効な各8bitを1つの画素データに変換する。

【0043】(1r、1g、1bはinteger型、 $0 \leq 1r, 1g, 1b \leq 255$)

前記の様に本発明のマイクロプロセッサにおける高速なpack処理では、floatingレジスタでfloating型のデータをinteger型のデータへ変換し、従来のpack処理では必要としていたfloatingレジスタからgeneralレジスタへのデータ転送処理を行わずに、そのままfloatingレジスタでbit演算処理を行い画素データに変換する。これにより、floatingレジスタからgeneralレジスタへのデータ転送処理が不要となり、pack処理を著しく高速化することができる。

【0044】次に、unpack処理について説明する。

【0045】図7に、本発明のマイクロプロセッサにおける高速なunpack処理のアセンブリ言語によるプログラムとレジスタの状態の例を示す。

【0046】(1)初めに、L702では、初期状態として、RGBの各成分1r、1g、1b各8bitを既にpack処理した画素データがfloatingレジスタ19番にある(702)。

【0047】(1r、1g、1bはinteger型、 $0 \leq 1r, 1g, 1b \leq 255$)

(2)次に、L705では、bit演算処理を行ってinteger型のデータをfloating型のデータへ変換する命令fextrucvxfにより、前記floatingレジスタ19番にある画素データのデータ1rの8bitをfloatingレジスタ16番に設定するbit演算処理を前記ALU(13)で行い、更に、前記データ1rをfloating型のデータrへ変換するデータ型の変換処理を前記加算器(11)で行う(705)。同様に、L706、707では、前記データ1g、1bについて同様な処理を行う(706、707)。この様にして、前記画素データの各成分1r、1g、1bの各8bitを、各成分を示す複数のfloating型のデータr、g、bへ変換する。

【0048】(r、g、bはfloating型、 $0.0 \leq r, g, b \leq 255.0$)

前記の様に本発明のマイクロプロセッサにおける高速なunpack処理では、従来のunpack処理では必要としていたgeneralレジスタからfloatingレジスタへのデータ転送処理を行わずに、そのままfloatingレジスタで画素データのbit演算処理を行い、integer型のデータをfloating型のデータへ変換する。これにより、generalレジスタからfloatingレジスタへのデータ転送処理が不要となり、unpack処理を著しく高速化することができる。

【0049】次に、別のFU(28)の構成について詳細に説明する。

【0050】図1に示した本発明のマイクロプロセッサのFU(10)は、加算器(11)、乗算器(12)、ALU(13)を1組ずつ有しているが、図3に示したように複数組有しても良い。

【0051】図3は、加算器、乗算器、ALUを3組有する本発明のマイクロプロセッサのFUとFRsの一実施例を示す構成図である。

【0052】FU(310)は、加減算やデータ型の変換などの演算を行う加算器(301、302、303)、乗除算や平方根などの演算を行う乗算器(304、305、306)、pack処理、unpack処理に必要なbit演算処理を行うALU(307、308、309)から構成される。FU(310)は、命令に従って、FRs(311)のデータを読み込み、加算器(301、302、303)、乗算器(304、305、306)、ALU(307、308、309)で演算を行い、その結果のデータをFRs(311)に書き込む。

【0053】次に、本発明のマイクロプロセッサの別のFUの構成における高速なpack処理とunpack処理について詳細に説明する。

【0054】図8に、本発明のマイクロプロセッサにおける高速なpack処理のアセンブリ言語によるプログラムとレジスタの状態の例を示す。

【0055】(1) 初めに、L802, 803, 804では、初期状態として、RGBの各成分を示す複数のfloating型のデータr, g, bがfloatingレジスタ16, 17, 18番にある(802, 803, 804)。

【0056】(r, g, bはfloating型, $0.0 \leq r, g, b \leq 255.0$)

(2) 次に、L807では、3つのfloating型のデータをinteger型のデータへ変換してbit 演算処理を行う命令fcvnxpack3により、前記floatingレジスタ16, 17, 18番にあるデータr, g, bをinteger型のデータIr, Ig, Ibへ変換するデータ型の変換処理を前記加算器(301, 302, 303)で行い、更に、前記データIr, Ig, Ibの有効な各8bitをfloatingレジスタ19番の所定の8bitの部分に設定するbit 演算処理を前記ALU(307, 308, 309)で行う(607)。この様にして、floating型のデータr, g, bをinteger型のデータIr, Ig, Ibに変換し、integer型のデータIr, Ig, Ibの有効な各8bitを1つの画素データに変換する。

【0057】(Ir, Ig, Ibはinteger型, $0 \leq Ir, Ig, Ib \leq 255$)

前記の様に本発明のマイクロプロセッサにおける高速なpack処理では、floatingレジスタでfloating型のデータをinteger型のデータへ変換し、従来のpack処理では必要としていたfloatingレジスタからgeneralレジスタへのデータ転送処理を行わずに、更に、前記実施例ではそのままfloatingレジスタで順次3回必要としていたbit 演算処理を1回のbit 演算処理で行い画素データに変換する。これにより、floatingレジスタからgeneralレジスタへのデータ転送処理が不要となり、更に、bit 演算処理を1回に削減しているため、pack処理を著しく高速化することができる。

【0058】図9に、本発明のマイクロプロセッサにおける高速なunpack処理のアセンブリ言語によるプログラムとレジスタの状態の例を示す。

【0059】(1) 初めに、L902では、初期状態として、RGBの各成分Ir, Ig, Ib各8bitを既にpack処理した画素データがfloatingレジスタ19番にある(902)。

【0060】(Ir, Ig, Ibはinteger型, $0 \leq Ir, Ig, Ib \leq 255$)

(2) 次に、L905では、3回分のbit 演算処理を行ってfloating型のデータをinteger型のデータへ変換する命令funpack3cnvxfにより、前記floatingレジスタ19番にある画素データのデータIr, Ig, Ibの各8bitをfloatingレジスタ16, 17, 18番に設定するbit 演算処理を前記ALU(307, 308, 309)

で行い、更に、前記データIr, Ig, Ibをfloating型のデータr, g, bへ変換するデータ型の変換処理を前記加算器(301, 302, 303)で行う(906, 907, 908)。この様にして、前記画素データの各成分Ir, Ig, Ibの各8bitを、各成分を示す複数のfloating型のデータr, g, bへ変換する。

【0061】(r, g, bはfloating型, $0.0 \leq r, g, b \leq 255.0$)

前記の様に本発明のマイクロプロセッサにおける高速なunpack処理では、従来のunpack処理では必要としていたgeneralレジスタからfloatingレジスタへのデータ転送処理を行わずに、更に、前記実施例ではそのままfloatingレジスタで順次3回必要としていたbit 演算処理を1回のbit 演算処理で画素データのbit 演算処理を行い、integer型のデータをfloating型のデータへ変換する。これにより、generalレジスタからfloatingレジスタへのデータ転送処理が不要となり、更に、bit 演算処理を1回に削減しているため、unpack処理を著しく高速化することができる。

【0062】以上、詳細に説明した様に、本発明のマイクロプロセッサは、グラフィックス・システムにおける画素データのpack処理、unpack処理において、floatingレジスタとgeneralレジスタ間のデータ転送処理を不要にし、更に、3回分のbit 演算処理を1回のbit 演算処理に削減している。これにより、本発明のマイクロプロセッサは、pack処理、unpack処理を著しく高速化することができる。

【0063】又、本発明のマイクロプロセッサをCPUに用いたグラフィックス・システムも、グラフィックス・システムにおける画素データのpack処理、unpack処理を著しく高速化することができる。

【0064】

【発明の効果】以上、詳細に説明した様に、本発明のグラフィックス・システムに適したマイクロプロセッサ、及びそれを適用したグラフィックス・システムによれば、グラフィックス・システムにおける画素データに関するpack処理、unpack処理を著しく高速化することができるという特有の効果を奏する。

【図面の簡単な説明】

【図1】本発明のマイクロプロセッサのFUとFRsの一実施例を示す構成図である。

【図2】本発明のマイクロプロセッサの一実施例を示す構成図である。

【図3】本発明のマイクロプロセッサの一実施例を示す構成図である。

【図4】従来のpack処理のアセンブリ言語によるプログラムとレジスタの状態を示す図である。

【図5】従来のunpack処理のアセンブリ言語によるプログラムとレジスタの状態を示す図である。

【図6】本発明のマイクロプロセッサにおける高速なpa

11

12

ck処理のアセンブリ言語によるプログラムとレジスタの状態を示す図である。

【図7】本発明のマイクロプロセッサにおける高速なunpack処理のアセンブリ言語によるプログラムとレジスタの状態を示す図である。

【図8】本発明のマイクロプロセッサにおける高速なpack処理のアセンブリ言語によるプログラムとレジスタの状態を示す図である。

【図9】本発明のマイクロプロセッサにおける高速なunpack処理のアセンブリ言語によるプログラムとレジスタ

の状態を示す図である。

【図10】本発明のグラフィックス・システムの一実施例を示す構成図である。

【符号の説明】

10…FU、11…加算器、12…乗算器、13…ALU、14、27、311…FRs、21…System Interface、22…IC、23…DC、24…Controller、25…GRs、26…IU、28、310…FU、301、302、303…加算器、304、305、306…乗算器、307、308、309…ALU。

【図1】

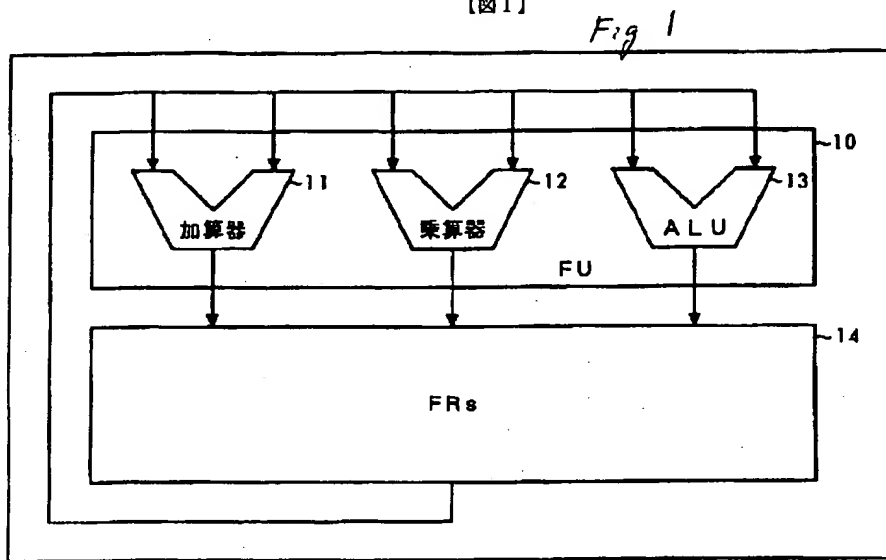
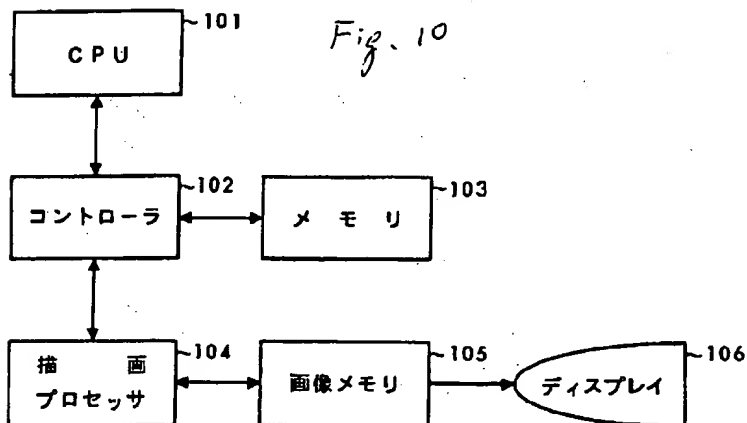


図
一

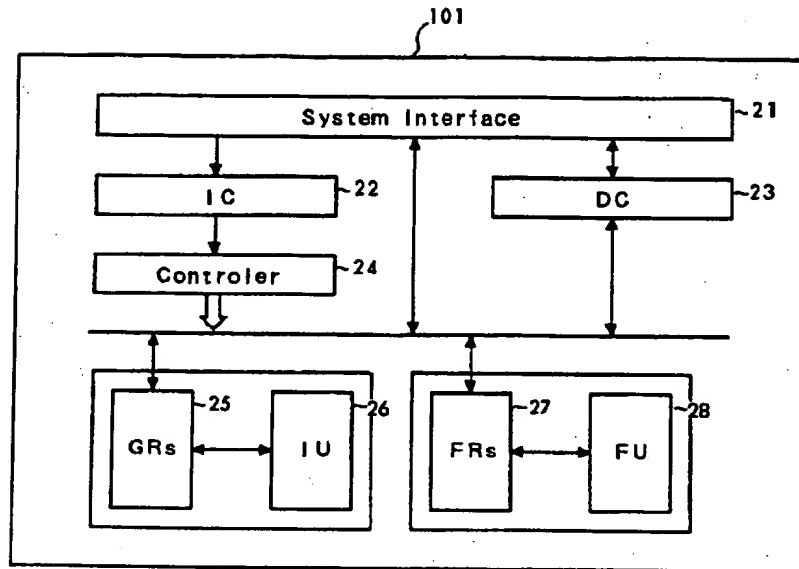
【図10】

図 10

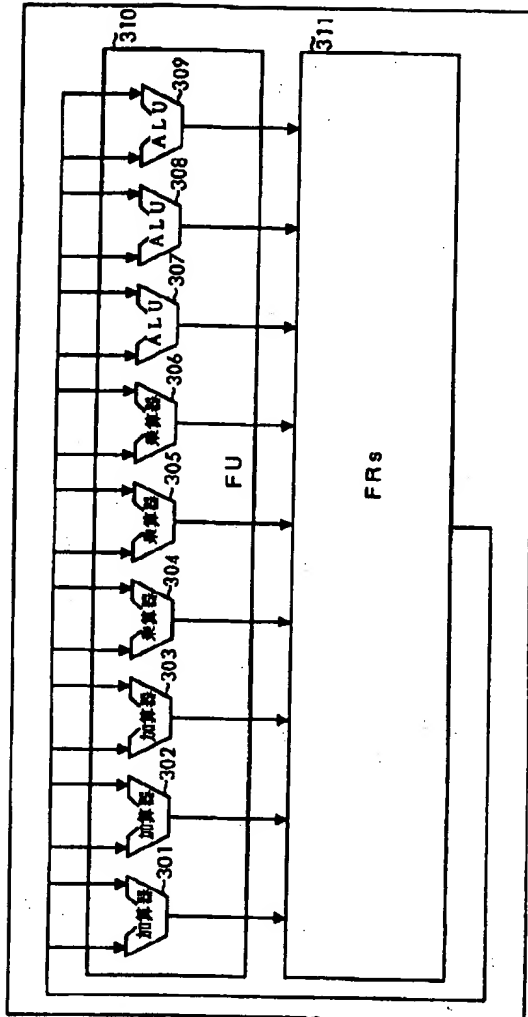


【図2】

図 2 FIG. 2



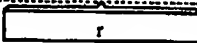
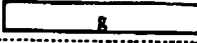
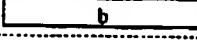
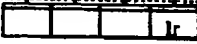
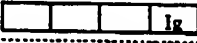

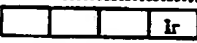

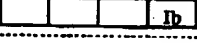
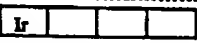

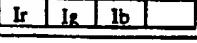
【図3】 Fig. 3
図 3



【図7】 Fig. 7
図 7

行 番号	プログラム	レジスタの状態
L701	/* 初期状態: %fr19はlr,lg,lbの各8bitを	8bit 8bit 8bit 8bit ~702
L702	packした画素データ*/	%fr19 = lr lg lb
L703	/* %fr19をunpackし、	
L704	integer型からfloating型へ変換*/	32bit
L705	fextrucnvxf %fr19, 7.8, %fr16 /* %fr16 = r*/	%fr16 = r ~705
L706	fextrucnvxf %fr19, 15.8, %fr17 /* %fr17 = g*/	%fr17 = g ~706
L707	fextrucnvxf %fr19, 23.8, %fr18 /* %fr18 = b*/	%fr18 = b ~707

【図4】 Fig-4
図 4

行 番号	プログラム	レジスタの状態
L401	/* 初期状態 */	32bit
L402	/* %fr16 = r */	%fr16 =  ~402
L403	/* %fr17 = g */	%fr17 =  ~403
L404	/* %fr18 = b */	%fr18 =  ~404
L405	/* floating型をinteger型へ変換 */	8bit 8bit 8bit 8bit
L406	fcvfix %fr16,%fr16 /* %fr16 = lr */	%fr16 =  ~406
L407	fcvfix %fr17,%fr17 /* %fr17 = lg */	%fr17 =  ~407
L408	fcvfix %fr18,%fr18 /* %fr18 = lb */	%fr18 =  ~408
L409	/* floatingレジスタからgeneralレジスタへ転送 */	
L410	fnogr %fr16,%gr16 /* %gr16 = lr */	%gr16 =  ~410
L411	fnogr %fr17,%gr17 /* %gr17 = lg */	%gr17 =  ~411
L412	fnogr %fr18,%gr18 /* %gr18 = lb */	%gr18 =  ~412
L413	/* lr,lg,lb各8bitを%gr19にpack処理する */	
L414	dep %gr16,7,8,%gr19	%gr19 =  ~414
L415	dep %gr17,15,8,%gr19	%gr19 =  ~415
L416	dep %gr18,23,8,%gr19	%gr19 =  ~416

【図5】

Fig. 5

図 5

行 番号	プログラム	レジスタの状態				
L501	/* 初期状態: %gr19はlr,lg,lbの各8bitを	8bit 8bit 8bit 8bit				
L502	packした画素データ */	%gr19 = <table><tr><td>lr</td><td>lg</td><td>lb</td><td></td></tr></table> ~502	lr	lg	lb	
lr	lg	lb				
L503	/* %gr19をunpack処理する */					
L504	extru %gr19,7,8,%gr16 /* %gr16 = lr */	%gr16 = <table><tr><td></td><td></td><td></td><td>lr</td></tr></table> ~504				lr
			lr			
L505	extru %gr19,15,8,%gr17 /* %gr17 = lg */	%gr17 = <table><tr><td></td><td></td><td></td><td>lg</td></tr></table> ~505				lg
			lg			
L506	extru %gr19,23,8,%gr18 /* %gr18 = lb */	%gr18 = <table><tr><td></td><td></td><td></td><td>lb</td></tr></table> ~506				lb
			lb			
L507	/* generalの31からfloatingの31へ転送 */					
L508	grtofr %gr16,%fr16 /* %fr16 = lr */	%fr16 = <table><tr><td></td><td></td><td></td><td>lr</td></tr></table> ~508				lr
			lr			
L509	grtofr %gr17,%fr17 /* %fr17 = lg */	%fr17 = <table><tr><td></td><td></td><td></td><td>lg</td></tr></table> ~509				lg
			lg			
L510	grtofr %gr18,%fr18 /* %fr18 = lb */	%fr18 = <table><tr><td></td><td></td><td></td><td>lb</td></tr></table> ~510				lb
			lb			
L511	/* integer型をfloating型へ変換 */	32bit				
L512	fcvxf %fr16,%fr16 /* %fr16 = r */	%fr16 = <table><tr><td></td><td>r</td></tr></table> ~512		r		
	r					
L513	fcvxf %fr17,%fr17 /* %fr17 = g */	%fr17 = <table><tr><td></td><td>g</td></tr></table> ~513		g		
	g					
L514	fcvxf %fr18,%fr18 /* %fr18 = b */	%fr18 = <table><tr><td></td><td>b</td></tr></table> ~514		b		
	b					

[図6]

図 6







Fig. 6

行 番号	プログラム	レジスタの状態
L601	/* 初期状態 */	
L602	/* %fr16 = r */	<div> <div>32bit</div> <div>%fr16 = r ~ 602</div> </div>
L603	/* %fr17 = g */	<div> <div>%fr17 = g ~ 603</div> </div>
L604	/* %fr18 = b */	<div> <div>%fr18 = b ~ 604</div> </div>
L605	/* floating型をinteger型へ変換し、	
L606	Ir,Ig,Ib各8bitを%fr19にpack処理する */	<div> <div>8bit 8bit 8bit 8bit</div> <div>%fr19 = Ir ~ 607</div> </div>
L607	fcvfxdep %fr16, 7,8,%fr19	
L608	fcvfxdep %fr17,15,8,%fr19	<div> <div>%fr19 = Ir Ig ~ 608</div> </div>
L609	fcvfxdep %fr18,23,8,%fr19	<div> <div>%fr19 = Ir Ig Ib ~ 609</div> </div>

【図8】

図 8







Fig. 8

行 番号	プログラム	レジスタの状態
L801	/* 初期状態 */	
L802	/* %fr16 = r */	32bit %fr16 =  ~802
L803	/* %fr17 = g */	%fr17 =  ~803
L804	/* %fr18 = b */	%fr18 =  ~804
L805	/* floating型をinteger型へ変換し、	
L806	lr,lg,lb各8bitを%fr19にpack処理する */	
L807	fcvxfpack3 %fr16,%fr17,%fr18,%fr19	8bit 8bit 8bit 8bit %fr19 =    ~807

【図9】

図 9

Fig. 9

行 番号	プログラム	レジスタの状態
L901	/* 初期状態 : %fr19は lr,lg,lbの各8bitを	
L902	packした画素データ */	8bit 8bit 8bit 8bit %fr19 =    ~902
L903	/* %fr19をunpack処理し、	
L904	integer型からfloating型へ変換 */	
L905	funpack3cvxf %fr19,%fr16,%fr17,%fr18	
L906	/* %fr16 = r */	32bit %fr16 =  ~906
L907	/* %fr17 = g */	%fr17 =  ~907
L908	/* %fr18 = b */	%fr18 =  ~908

フロントページの続き

(72)発明者 藤井 秀樹

茨城県日立市大みか町五丁目2番1号 株
式会社日立製作所大みか工場内

**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ **BLACK BORDERS**
- ☐ **IMAGE CUT OFF AT TOP, BOTTOM OR SIDES**
- ☒ **FADED TEXT OR DRAWING**
- ☐ **BLURRED OR ILLEGIBLE TEXT OR DRAWING**
- ☐ **SKEWED/SLANTED IMAGES**
- ☐ **COLOR OR BLACK AND WHITE PHOTOGRAPHS**
- ☐ **GRAY SCALE DOCUMENTS**
- ☒ **LINES OR MARKS ON ORIGINAL DOCUMENT**
- ☐ **REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY**
- ☐ **OTHER:** _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.